Project 1 – Final Specification

April 14, 2022

1 Distributed Memory Advection

Due Date: Wednesday, April 20, 2022 by 11:59 pm

To complete Project 1, you must include the following functionality, using your Milestone 2 implementation as a starting point, but creating a new Github directory *final-version*.

- Include a README with your name, the assignment, a list of any references you used, and a discussion of any shortcomings your code may have. You must also provide a Makefile and instructions for compiling and running your code. (5 points)
- Lax serial solver with periodic boundaries, as required in Milestone 1 (15 points)
- Support for on node scaling using OpenMP, as required in Milestone 2 (15 points)
- Support for hydbrid parralelism using MPI (35 points)
- Support for 1st order upwind scheme in full version (10 points)
- Support for non-uniform u,v in full version (10 points)
- Support for 2nd order upwind scheme in full version (10 points)

All of the this functionality needs to be developed, tested, benchmarked, and reported in your final writeup, which must contain the following sections:

I. Serial Solver with constant u,v and periodic boundaries
   i. Brief description of algorithm, implementation
   ii. Demonstration of correctness
   iii. Performance analysis

II. OpenMP parallel solver
   i. Brief description of implementation
   ii. Demonstration of correctness
   iii. Performance analysis

III. Hybrid MPI/OpenMP solver
   i. Description of parallel strategy. Note that following requirements
      - Use square domain with square decomposition as explained in supplemental video. Can assume number n^2 MPI ranks with N divisible by n.
      - C and $Cnew$ data structures must be decomposed across MPI ranks – no single rank can allocate the entire data structure.

- Input parameters must be read on rank zero and broadcast (when needed) to other ranks.
- Parallel i/o strategy is your choice but must be explained.

ii. Demonstration of correctness. This should include a movie or sequence of plots qualitatively verifying reasonable behavior. Also comment on expectations for bitwise reproducibility for your implementation.

iii. Performance analysis. Report the the following:

- strong scaling: parallel efficiency and grind rate for the $N = 10000$ benchmark using 1,4 and 16 nodes, 1 MPI rank per node, and the optimal number of OpenMP threads per rank.
- weak scaling: same as above but weak scaling a local (per node) problem size of $N = 10000$.
- Redo the above using 1 MPI rank per core with and no use of OpenMP

IV. Add first-order upwind scheme with brief explanation of algorithm and demonstration of correctness (you may assume $\Delta x = \Delta y$).

- First order upwind: for $u, v > 0$

$$\frac{C^{n+1}_{i,j} - C^n_{i,j}}{\Delta t} + u\frac{C^n_{i,j} - C^n_{i-1,j}}{\Delta x} + v\frac{C^n_{i,j} - C^n_{i,j-1}}{\Delta y} = 0$$

- for $u, v < 0$

$$\frac{C^{n+1}_{i,j} - C^n_{i,j}}{\Delta t} + u\frac{C^n_{i+1,j} - C^n_{i,j}}{\Delta x} + v\frac{C^n_{i,j+1} - C^n_{i,j}}{\Delta y} = 0$$

V. Add support for non-uniform (u,v) with a particular choice of (u, v). (u, v) must be chosen so that it is solenoidal (divergence-free) to satisfy the conservation law.

i. describe choice of (u,v)

ii. show movie of a sequence of images of solution

VI. Add 2nd order upwind scheme with brief explanation of algorithm and demonstration of correctness. Note this requires 2 levels of ghost cells and should be handled using good abstractions in your code.

- Second order upwind for $u, v > 0$

$$\frac{C^{n+1}_{i,j} - C^n_{i,j}}{\Delta t} + u\frac{3C^n_{i,j} - 4C^n_{i-1,j} + C^n_{i-2,j}}{\Delta x} + v\frac{3C^n_{i,j} - 4C^n_{i,j-1} + C^n_{i,j-2}}{\Delta y} = 0$$

- for $u, v < 0$

$$\frac{C^{n+1}_{i,j} - C^n_{i,j}}{\Delta t} + u\frac{-C^n_{i,j} + 4C^n_{i-1,j} - 3C^n_{i-2,j}}{\Delta x} + v\frac{-C^n_{i,j} + 4C^n_{i,j-1} - 3C^n_{i,j-2}}{\Delta y} = 0$$